"For even the very wise cannot see all ends."

J.R.R. Tolkien, The Fellowship of the Ring

**CMPUT 365
Introduction to RL**

Marlos C. Machado

Coursera Reminder

You **should be enrolled in the private session** we created in Coursera for CMPUT 365.

I **cannot** use marks from the public repository for your course marks.

You **need** to **check**, **every time**, if you are in the private session and if you are submitting quizzes and assignments to the private section.

The deadlines in the public session **do not align** with the deadlines in Coursera.

If you have any questions or concerns, **talk with the TAs** or email us
cmput365@ualberta.ca.

Reminders and Notes

- The practice quiz is due today.

- The programming assignment is due on Wednesday.

- The grades for Module 2 of Coursera are now available on eClass. If you have any questions / concerns, please contact <u>cmput365@ualberta.ca</u>.

- Midterm 2 is marked. Grades should be up today or tomorrow.

    - Exam viewing will be on Wednesday and Thursday.

    - Wednesday from 10am to 1pm at CSC 3-49.

    - Thursday from 1pm to 4pm at CSC 3-50.

Marlos C. Machado

Please, interrupt me at any time!

Part II:
# Approximate Solution Methods

Marlos C. Machado

What We Have Done So Far: *Tabular RL*

- Absolutely everything we did was in the *tabular* case.

  - There's a huge memory cost in having to fill a table with a ridiculous total number of states.

  - We might never see the same state twice.

  - We cannot expect to find an optimal policy (or value function) even in the limit of infinite time and data.

- What about…
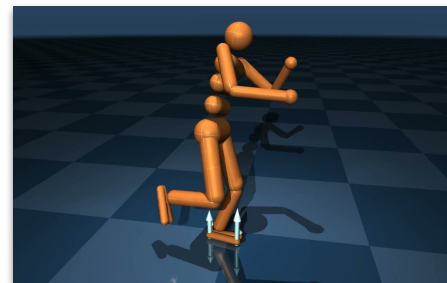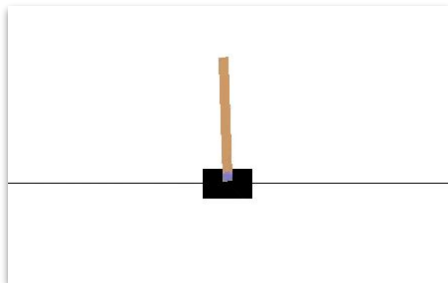
From Now On: *Generalization*

- Instead, we should find a good approximate solution using limited computational resources.

- We need to generalize to from previous encounters with different states that are in some sense similar to the current one.

- We obtain generalization with *function approximation* (often from the supervised learning literature).

em?
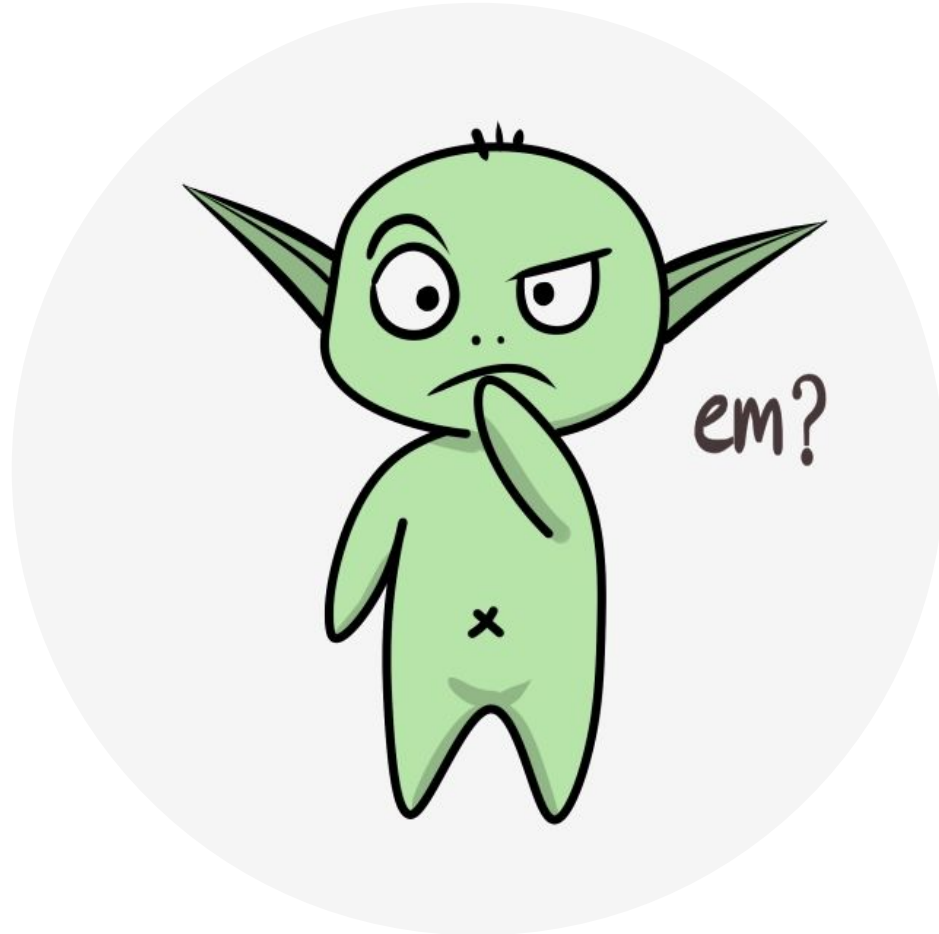
Function Approximation – An Example
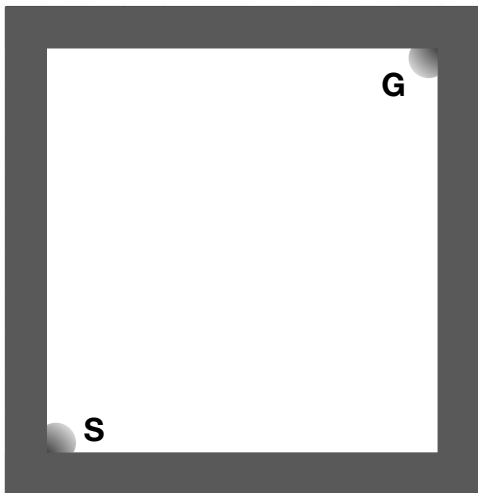
State space: $<x, y>$ coordinates (continuous, no grid) and $<\dot{x}, \dot{y}>$ velocity (continuous).

Start state: Somewhere in the bottom left corner, where a suitable $<x, y>$ coordinate is selected randomly.

Action space: Adding or subtracting a small force to $\dot{x}$ velocity or $\dot{y}$ velocity, or leaving them unchanged.

Dynamics: Traditional physics + noise, collisions with obstacles are fully elastic and cause the agent to bounce.
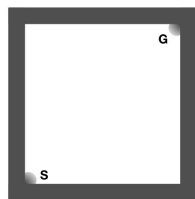
Reward function: +1 when you hit the region in G.

γ: 0.9.

Function Approximation

- In the tabular case, $\mathbf{v}_\pi \in \mathbb{R}^{|\mathscr{S}|}$.

- Instead, we will approximate $\mathbf{v}_\pi$ using a function parameterized by some weights $\mathbf{w} \in \mathbb{R}^d$ where $d \ll |\mathscr{S}|$. We will write $\hat{v}(s, \mathbf{w}) \approx v_\pi(s)$.

Function Approximation

- In the tabular case, $\mathbf{v}_\pi \in \mathbb{R}^{|\mathscr{S}|}$.

- Instead, we will approximate $\mathbf{v}_\pi$ using a function parameterized by some weights $\mathbf{w} \in \mathbb{R}^d$ where $d \ll |\mathscr{S}|$. We will write $\hat{v}(s,\mathbf{w}) \approx v_\pi(s)$.

- An example:

$$\mathbf{s} = \begin{bmatrix} x \\ y \\ \dot{x} \\ \dot{y} \end{bmatrix} \qquad \mathbf{w} = \begin{bmatrix} w_1 \\ w_2 \\ w_3 \\ w_4 \end{bmatrix} \qquad \hat{v}\,(\mathbf{s}, \mathbf{w}) = \mathbf{s}^\top \mathbf{w}$$

Generalization: When something changes, many states can be affected.

Feature vector

Function Approximation

- In the tabular case, $\mathbf{v}_\pi \in \mathbb{R}^{|\mathcal{S}|}$.

- Instead, we will approximate $\mathbf{v}_\pi$ using a function parameterized by some weights $\mathbf{w} \in \mathbb{R}^d$ where $d \ll |\mathcal{S}|$. We will write $\hat{v}(s,\mathbf{w}) \approx v_\pi(s)$.
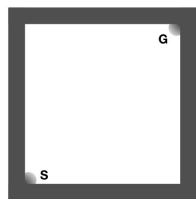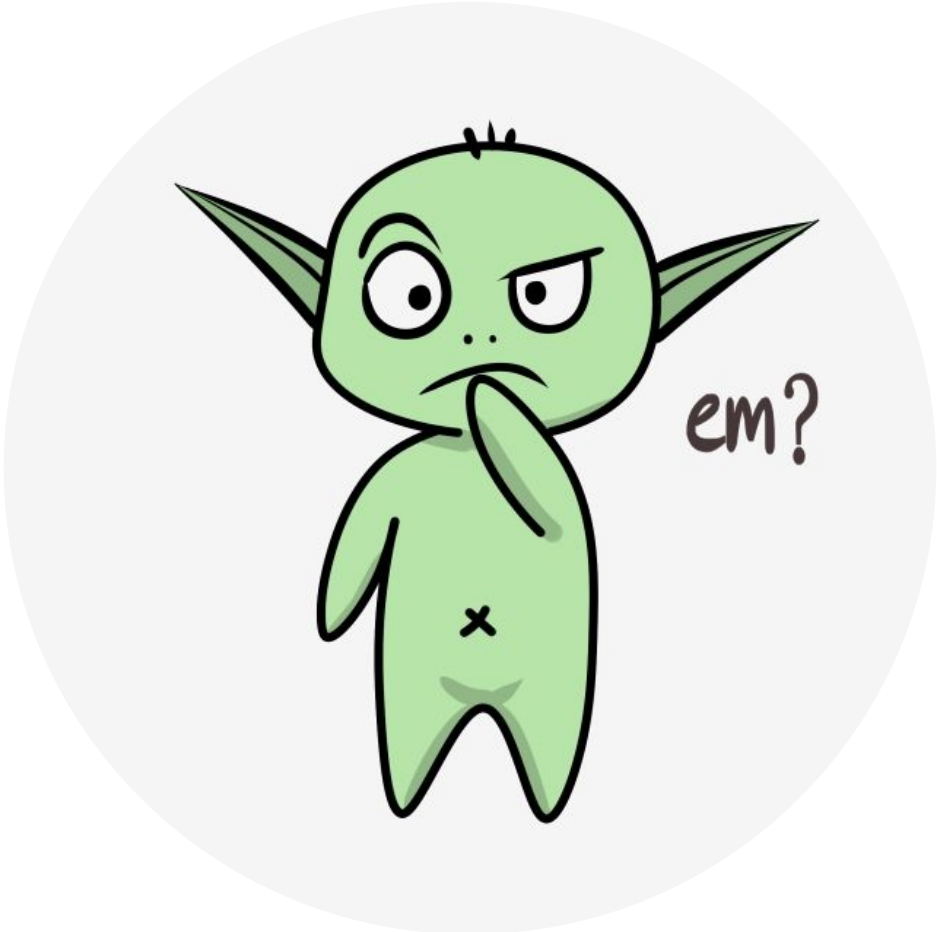
- An example:

$$\mathbf{s} = \begin{bmatrix} x \\ y \\ \dot{x} \\ \dot{y} \end{bmatrix} \qquad \mathbf{w} = \begin{bmatrix} w_1 \\ w_2 \\ w_3 \\ w_4 \end{bmatrix} \qquad \hat{v}(\mathbf{s}, \mathbf{w}) = \mathbf{s}^\top \mathbf{w}$$

- Extending RL to function approximation also makes it applicable to partially observable problems, in which the full state is not available to the agent. I may use $\mathbf{o}$ to denote the agent's observation (instead of $\mathbf{s}$).

em?

Chapter 9

On-policy Prediction with Approximation

Prediction

Value-function Approximation

- We can interpret each update we have seen so far as an example of the desired input-output behavior of the value function.

- Let $s \mapsto u$ denote an individual update, where s is the state updated and u the update target that s's estimated value is shifted to.
  - $S_t \mapsto G_t$ **Monte Carlo update**
  - $S_t \mapsto R_{t+1} + \gamma\, v_\pi(S_{t+1})$ **TD(0)**
  - $s \mapsto \mathbb{E}_\pi [R_{t+1} + \gamma\, v_\pi(S_{t+1}) \mid S_t = s]$ **Dynamic Programming**

- Supervised learning methods learn to mimic input-output examples, and when the output are numbers, the process is called *function approximation*.

A note from the textbook

Viewing each update as a conventional training example in this way enables us to use any of a wide range of existing function approximation methods for value prediction. In principle, we can use any method for supervised learning from examples, including artificial neural networks, decision trees, and various kinds of multivariate regression. However, not all function approximation methods are equally well suited for use in reinforcement learning. The most sophisticated artificial neural network and statistical methods all assume a static training set over which multiple passes are made. In reinforcement learning, however, it is important that learning be able to occur online, while the agent interacts with its environment or with a model of its environment. To do this requires methods that are able to learn efficiently from incrementally acquired data. In addition, reinforcement learning generally requires function approximation methods able to handle nonstationary target functions (target functions that change over time). For example, in control methods based on GPI (generalized policy iteration) we often seek to learn q_π while π changes. Even if the policy remains the same, the target values of training examples are nonstationary if they are generated by bootstrapping methods (DP and TD learning). Methods that cannot easily handle such nonstationarity are less suitable for reinforcement learning.

Not really!

Marlos C. Machado

The Prediction Objective (A Notion of Accuracy)

- In the tabular case we can have equality, but with FA, not anymore.
  - Making one state's estimate more accurate invariably means making others' less accurate.

- Mean Squared Error:

$$\overline{\text{VE}}(\mathbf{w}) \doteq \sum_{s \in \mathcal{S}} \mu(s) \Big[v_\pi(s) - \hat{v}(s, \mathbf{w}) \Big]^2$$

How much do we care about the error in each state s.

Usually, the fraction of time spent in s.
On-policy distribution.

Marlos C. Machado

The Prediction Objective (A Notion of Accuracy)

- In the tabular case we can have equality, but with FA, not anymore.
  - Making one state's estimate more accurate invariably means making others' less accurate.

- Mean Squared Error:

$$\overline{\text{VE}}(\mathbf{w}) \doteq \sum_{s \in \mathcal{S}} \mu(s) \Big[v_\pi(s) - \hat{v}(s, \mathbf{w}) \Big]^2$$

How much do we care about the error in each state s.

Usually, the fraction of time spent in s.
On-policy distribution.

- When doing nonlinear function approximation, we lose pretty much every guarantee we had (often, even convergence guarantees).

Marlos C. Machado

em?

21

Question

How are tabular methods related to linear function approximation?

Marlos C. Machado